

分布式系统 HDFS 部署报告

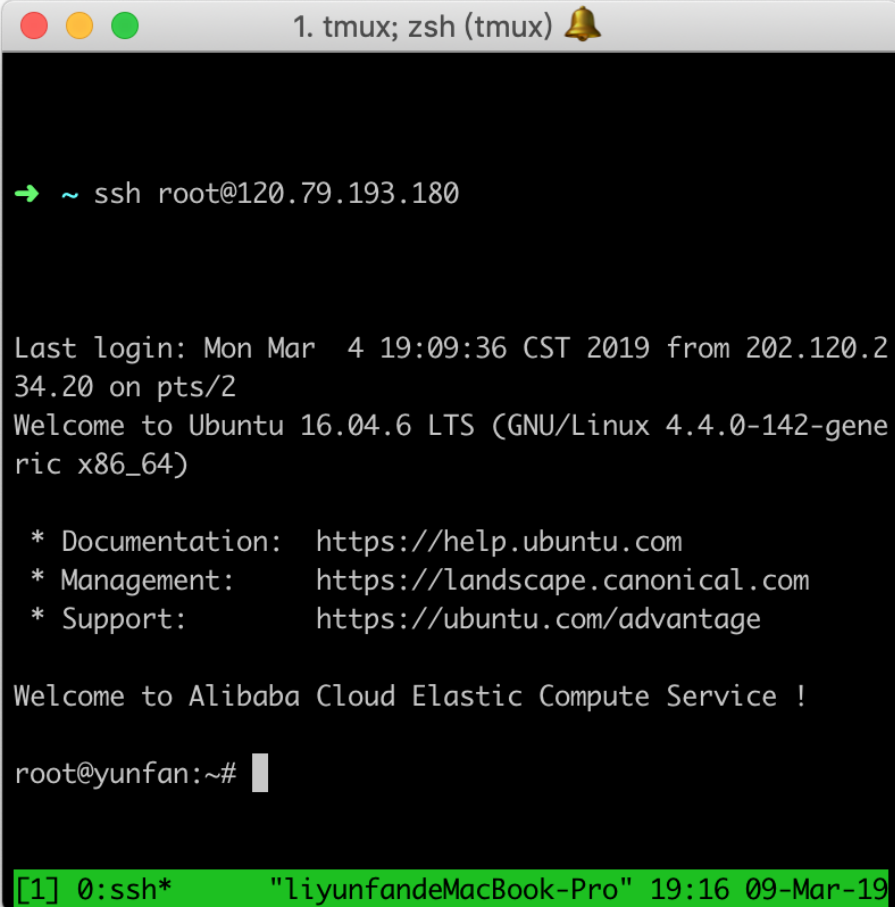
16302010002 李云帆

部署环境：阿里云，Ubuntu16.04 事先已经安装过 JDK 并配置过公钥登陆

（由于以前已经部署好了，所以现在是依靠回忆来回放一个部署过程，当时部署时

参照了[一篇博客](#)，助教检查的时候可以看我的网站

<http://985lovestory.online:9870/>)



```
1. tmux; zsh (tmux) 🔔

➔ ~ ssh root@120.79.193.180

Last login: Mon Mar  4 19:09:36 CST 2019 from 202.120.234.20 on pts/2
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-142-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Welcome to Alibaba Cloud Elastic Compute Service !

root@yunfan:~#
```

[1] 0:ssh* "liyunfandeMacBook-Pro" 19:16 09-Mar-19

1. 第一步，登上阿里云服务器。
2. 按照网上教程设置公钥（免密）登陆，
3. 安装 JDK，
4. 下载安装 Hadoop 3 ([网址](#))，一般选择下载最新的稳定版本，将下载的 tar.gz 保存到 root 目录下，使用命令

```
sudo tar -zxvf PATH/TO/hadoop*.tar.gz -C /usr/local
```

将其解压到自定义的文件夹下，我是放在/opt/software/hadoop 下)，

5. Hadoop 解压后即可使用。输入

```
cd PATH/TO/hadoop  
./bin/hadoop version
```

命令来检查 Hadoop 是否可用，成功则会显示 Hadoop 版本信息：

6. Hadoop 伪分布式配置

伪分布式模式只需要改两个配置文件并且格式化 namenode 即可

编辑文件 etc/hadoop/core-site.xml：

```
<configuration>  
  <property>  
    <name>fs.defaultFS</name>  
    <value>hdfs://localhost:9000</value>  
  </property>  
</configuration>
```

编辑文件 etc/hadoop/hdfs-site.xml：

```
<configuration>
```

```
<property>
  <name>hadoop.tmp.dir</name>
  <value>file:/usr/local/src/hadoop-3/tmp</value>
  <description>Abase for other temporary
directories.</description>
</property>
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
<property>
  <name>dfs.namenode.name.dir</name>
  <value>file:/usr/local/src/hadoop-
3/tmp/dfs/name</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>file:/usr/local/src/hadoop-3/tmp/dfs/data</value>
</property>
</configuration>
```

Hadoop 配置文件说明：

Hadoop 的运行方式是由配置文件决定的（运行 Hadoop 时会读取配置文件）

配置完成后，执行 namenode 的格式化：

```
hdfs namenode -format
```

然后使用 start-dfs.sh 命令启动 NameNode daemon 进程和 DataNode daemon 进程：

在启动前需要修改 etc/hadoop/hadoop-env.sh 文件中的 JAVA_HOME 变量为实际的即可。

**启动完成后，可以通过命令 jps 来判断是否成功启动，若成功启动则会列出如下进程：“NameNode”、“DataNode”和
“SecondaryNameNode”**

要使用 HDFS，首先需要在 HDFS 中创建用户目录(我的用户就是 root)：

```
hdfs dfs -mkdir -p /user/root
```

接着将 etc/hadoop 中的 xml 文件作为输入文件复制到分布式文件系统中，即将 /opt/software/root/etc/hadoop 目录下的 xml 文件复制到分布式文件系统上的 /user/root/input 中。我们使用的是 root 用户，并且已创建相应的用户目录 /user/root，因此在命令中就可以使用相对路径如

input，其对应的绝对路径就是 /user/root/input：

```
hdfs dfs -mkdir input  
hdfs dfs -put ./etc/hadoop/*.xml input
```

复制完成后，可以通过如下命令查看文件列表：

```
hdfs dfs -ls input
```

伪分布式运行 MapReduce 作业的方式跟单机模式相同，区别在于伪分布式读取的是 HDFS 中的文件（可以将单机步骤中创建的本地 input 文件夹，输出结果 output 文件夹都删掉来验证这一点）。

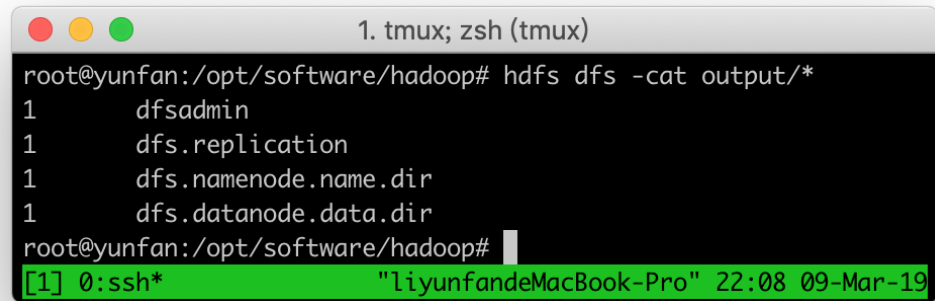
```
1. tmux; zsh (tmux)
root@yunfan:/opt/software/hadoop# hdfs dfs -ls input
Found 9 items
-rw-r--r-- 1 root supergroup      8260 2019-03-09 22:05 input/capacity-scheduler.xml
-rw-r--r-- 1 root supergroup      1147 2019-03-09 22:05 input/core-site.xml
-rw-r--r-- 1 root supergroup    11392 2019-03-09 22:05 input/hadoop-policy.xml
-rw-r--r-- 1 root supergroup      1135 2019-03-09 22:05 input/hdfs-site.xml
-rw-r--r-- 1 root supergroup       620 2019-03-09 22:05 input/https-site.xml
-rw-r--r-- 1 root supergroup     3518 2019-03-09 22:05 input/kms-acls.xml
-rw-r--r-- 1 root supergroup       682 2019-03-09 22:05 input/kms-site.xml
-rw-r--r-- 1 root supergroup       758 2019-03-09 22:05 input/mapred-site.xml
-rw-r--r-- 1 root supergroup       690 2019-03-09 22:05 input/yarn-site.xml
root@yunfan:/opt/software/hadoop# hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-3.2.0.jar gre
p input output 'dfs[a-z.]+'
2019-03-09 22:06:23,268 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2019-03-09 22:06:23,410 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2019-03-09 22:06:23,410 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2019-03-09 22:06:23,906 INFO input.FileInputFormat: Total input files to process : 9
2019-03-09 22:06:23,969 INFO mapreduce.JobSubmitter: number of splits:9
2019-03-09 22:06:24,205 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1305963494_0001
2019-03-09 22:06:24,207 INFO mapreduce.JobSubmitter: Executing with tokens: []
2019-03-09 22:06:24,435 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2019-03-09 22:06:24,435 INFO mapreduce.Job: Running job: job_local1305963494_0001
2019-03-09 22:06:24,440 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2019-03-09 22:06:24,450 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2019-03-09 22:06:24,450 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folder
s under output directory:false, ignore cleanup failures: false
2019-03-09 22:06:24,451 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapreduce.lib.outp
ut.FileOutputCommitter
2019-03-09 22:06:24,551 INFO mapred.LocalJobRunner: Waiting for map tasks
2019-03-09 22:06:24,552 INFO mapred.LocalJobRunner: Starting task: attempt_local1305963494_0001_m_000000_0
2019-03-09 22:06:24,592 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2019-03-09 22:06:24,592 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folder
s under output directory:false, ignore cleanup failures: false
2019-03-09 22:06:24,632 INFO mapred.Task: Using ResourceCalculatorProcessTree : [ ]
2019-03-09 22:06:24,638 INFO mapred.MapTask: Processing split: hdfs://localhost:9000/user/root/input/hadoop
-policy.xml:0+11392
2019-03-09 22:06:24,791 INFO mapred.MapTask: (EQUATOR) 0 kvi 26214396(104857584)
2019-03-09 22:06:24,791 INFO mapred.MapTask: mapreduce.task.io.sort.mb: 100
[1] 0:[tmux]*"liyunfandeMacBook-Pro" 22:07 09-Mar-19
```

```
hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-3.x.jar grep input output 'dfs[a-z.]+'
```

查看运行结果的命令（查看的是位于 HDFS 中的输出结果）：

```
hdfs dfs -cat output/*
```

结果如下，注意到刚才我们已经更改了配置文件，所以运行结果不同。

A terminal window titled "1. tmux; zsh (tmux)" showing the execution of the command "hdfs dfs -cat output/*". The output lists four HDFS configuration parameters: "dfsadmin", "dfs.replication", "dfs.namenode.name.dir", and "dfs.datanode.data.dir", each preceded by a "1". The prompt "root@yunfan:/opt/software/hadoop#" is visible. A green status bar at the bottom shows "[1] 0:ssh*" and the system information "liyunfandeMacBook-Pro" 22:08 09-Mar-19.

```
1. tmux; zsh (tmux)
root@yunfan:/opt/software/hadoop# hdfs dfs -cat output/*
1      dfsadmin
1      dfs.replication
1      dfs.namenode.name.dir
1      dfs.datanode.data.dir
root@yunfan:/opt/software/hadoop#
[1] 0:ssh* "liyunfandeMacBook-Pro" 22:08 09-Mar-19
```

也可以将运行结果取回到本地：

```
./bin/hdfs dfs -get output ./output
cat ./output/*
```

Hadoop 运行程序时，输出目录不能存在，否则会提示错误

org.apache.hadoop.mapred.FileAlreadyExistsException: Output directory hdfs://localhost:9000/user/root/output already exists，因此

此若要再次执行，需要执行如下命令删除 output 文件夹：

```
hdfs dfs -rm -r output
```

若要关闭 Hadoop，则运行

```
stop-dfs.sh
```

下次启动 hadoop 时，无需进行 NameNode 的初始化，只需要运

行 **start-dfs.sh** 就可以！

对比 HDFS 与 NFS

NFS（网络文件系统）：开发的一种基于协议，允许客户端通过网络访问文件的文件系统。NFS 客户端允许用户访问文件，就像文件驻留在用户的本地设备上一样，实际它们存储在联网计算机的磁盘上。

HDFS（Hadoop 分布式文件系统）：分布在许多联网计算机或节点中的文件系统。HDFS 具有容错能力，因为它在文件系统中存储了多个文件副本。

最大的区别是 复本（容错性）。HDFS 旨在应对失败。NFS 没有内置任何容错功能。

除容错外，由于 HDFS 支持多个文件副本。这可以消除（或简化）许多客户端访问单个文件的常见瓶颈。同时由于文件具有多个副本，因此在不同的物理磁盘上，读取性能比 NFS 更好。